iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** V **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.82524

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

Nexora – Developing A Modern Social Networking Platform Using MERN Stack

Rakesh Kummari[1], Banesh Durgam[2], Aaryan Pawar[3], Jayesh Patil[4], Ajay Girase[5], Prof. Vishal R. Deshmukh[6]
Department of Computer Engineering
[1,2,3,4,5]*Shri Shivaji Vidya Prasarak Sanstha's Bapusaheb Shivajirao Deore College of Engineering, Dhule 424005, Maharashtra, India*

Abstract: ***Social networking platforms have become essential components of modern digital communication by enabling users to connect, share information, and build online communities. This research presents the design and development of Nexora, a scalable social networking platform built using the MERN stack, which includes MongoDB, Express.js, React.js, and Node.js. The platform provides secure user authentication, profile management, post creation, likes, comments, notifications, and responsive social interaction features. React.js delivers a dynamic frontend interface, while Node.js and Express.js manage backend services and RESTful APIs. MongoDB supports flexible storage of user-generated data. The system architecture emphasizes scalability, maintainability, and performance while supporting modern social networking requirements. Testing results demonstrate effective response times, reliable data management, and secure communication. This study concludes that MERN stack technology offers an efficient and practical solution for modern web-based social networking applications.***
Keywords: MERN Stack, Social Networking, React.js, Node.js, MongoDB, Express.js, REST API, Web Development, Real-Time Communication

I. INTRODUCTION

The proposed system, Nexora, is a modern full-stack responsive social networking platform developed using the MERN stack, which consists of MongoDB, Express.js, React.js, and Node.js [1], [2]. The architecture of the system is based on a structured three-tier model that ensures scalability, modularity, maintainability, and secure communication between system components [4], [5]. By implementing CRUD operations across all major functionalities, the platform delivers an efficient and reliable web-based environment capable of supporting modern social networking demands [3].

The system emphasizes secure user authentication and profile management through complete registration validation, protected login mechanisms, and profile image upload capabilities [2], [18]. Users are able to create personalized accounts, securely access the platform, and interact with a dynamic homepage that integrates multiple widgets, profile displays, and social content feeds. This design approach improves user engagement while maintaining responsive performance across multiple devices [13], [15].

Nexora incorporates essential social networking functionalities such as post creation, editing, deletion, image sharing, likes, dislikes, comments, and friend management systems. Users can interact with other profiles, manage social relationships, and access personalized feeds through an intuitive interface [3], [14]. Additional features including dark mode, light mode, and mobile responsiveness enhance accessibility, usability, and overall user experience [16].

The frontend implementation is exclusively developed using React.js, enabling reusable component-based architecture and efficient virtual DOM rendering [13]. Supporting technologies such as React Router, Formik, Yup, Redux Toolkit, Redux Persist, Material UI, and React Dropzone collectively improve navigation, validation, state management, styling, and media upload processes [1], [2]. This combination significantly enhances development efficiency and frontend scalability.

The backend infrastructure utilizes Node.js and Express.js for server-side execution, RESTful API development, and business logic management [9], [11]. MongoDB serves as the primary. Database for handling dynamic user-generated content, while Mongoose provides schema modeling and structured database operations [6], [7], [8]. Security is reinforced through JSON Web Token (JWT) authentication, Multer for media file handling, and middleware for HTTP request validation [18]. This backend architecture ensures secure communication, optimized performance, and efficient full-stack integration. The development of Nexora primarily aims to apply industrial software engineering practices in building a secure, scalable, and maintainable social networking platform while providing practical implementation experience in MERN stack technologies [5], [12]. The project successfully demonstrates how modern full-stack frameworks can be utilized to develop real-world web applications that support large-scale user interaction, dynamic content management, and future system scalability [4], [19].

II. LITERATURE SURVEY

Full-stack web development has gained significant importance in recent years due to the rapid growth of dynamic web applications and increasing demand for scalable digital platforms [1], [2].

A full-stack developer is responsible for managing both frontend and backend development processes, enabling the creation of complete web-based systems through integrated technologies. The rise of modern frameworks has simplified software development while improving efficiency, maintainability, and deployment speed [5].

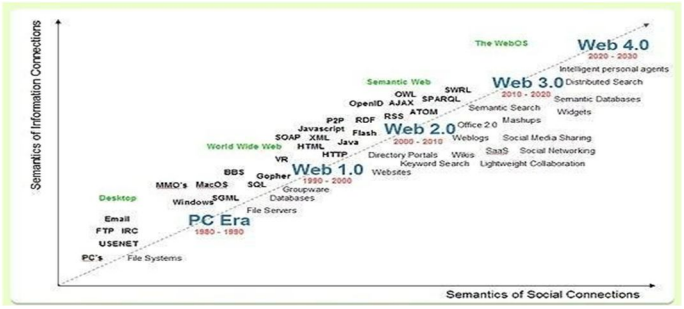


Fig. 2.1: The Evolution of Web

The evolution of web technologies has progressed through multiple generations, beginning with static Web 1.0 systems focused primarily on information display, followed by Web 2.0, which introduced interactive social platforms, blogs, and user-generated content [12].

Table 2.1: Evolution of Web Technologies

Web Generation	Features	Major Technologies
Web 1.0	Static websites, read-only content	HTML, HTTP
Web 2.0	Social networking, blogs, user interaction	JavaScript, CSS, AJAX
Web 3.0	Semantic web, intelligent search, personalization	RDF, OWL, XML
Web 4.0	Active web, AI integration, automation	AI, IoT, Cloud

The emergence of social networking platforms such as Facebook, Twitter, and LinkedIn transformed digital communication globally by enabling real-time social interaction and collaborative web experiences [10]. Web 3.0 further introduced semantic technologies, intelligent search systems, and personalized content delivery, while Web 4.0 continues to emphasize active web ecosystems, automation, and advanced user experiences [12].

Traditional web development initially relied on technologies such as the LAMP stack, which included Linux, Apache, MySQL, and PHP. However, the emergence of JavaScript-based full-stack ecosystems such as MERN (MongoDB, Express.js, React.js, Node.js) has revolutionized web development by enabling unified programming across both client-side and server-side environments [2], [4]. This unified architecture reduces development complexity, enhances maintainability, and improves scalability for modern applications [1].

MongoDB has become one of the most widely adopted NoSQL databases due to its flexibility, document-oriented structure, and ability to manage large-scale dynamic data efficiently [6], [7], [9]. Compared to traditional relational databases, MongoDB offers improved scalability, schema flexibility, and faster data retrieval, making it highly suitable for social networking platforms that require continuous user-generated content management [8].

Node.js introduced a powerful server-side JavaScript runtime capable of supporting asynchronous, event-driven programming for high-performance network applications [9]. Express.js further enhances Node.js by providing robust API development tools, middleware support, and efficient web server capabilities, enabling developers to create scalable and maintainable backend systems [11].

React.js has emerged as one of the leading frontend frameworks due to its reusable component-based architecture, Virtual DOM rendering, and ability to create responsive user interfaces efficiently [13]. Its flexibility and performance optimization capabilities make it highly suitable for modern web application development, especially for dynamic platforms requiring continuous UI updates [14], [15].

Web design and user experience have also evolved substantially, shifting from simple static interfaces to highly interactive, responsive, and user-friendly systems [16]. Modern frontend technologies such as React.js, Material UI, and component-based frameworks significantly improve interface scalability, maintainability, and accessibility across devices.

Several studies have highlighted that MERN stack technologies collectively provide a practical solution for building modern social networking platforms, e-commerce systems, and enterprise applications [3], [4]. The integration of React.js, Node.js, Express.js, and MongoDB creates a cohesive development ecosystem that improves development speed, security, and performance while supporting real-world application scalability.

Thus, the literature strongly supports the adoption of MERN stack technologies for modern full-stack development, particularly for applications requiring secure authentication, responsive interfaces, dynamic data handling, and future scalability. These advancements form the technological foundation for the development of Nexora as a modern social networking platform.

III. SYSTEM DEVELOPMENT

A. *Architecture Design*

The Nexora platform is developed using the MERN stack architecture, which integrates MongoDB, Express.js, React.js, and Node.js into a unified full-stack JavaScript ecosystem [1], [2]. The system follows a three-tier architecture consisting of the presentation layer (frontend), application layer (backend), and database layer. React.js is responsible for delivering a responsive and interactive user interface, while Node.js and Express.js manage backend logic, RESTful APIs, and server communication [4], [9]. MongoDB serves as the NoSQL database for storing dynamic user-generated content efficiently [6], [7]. This architecture ensures modularity, scalability, maintainability, and secure communication throughout the platform [5].

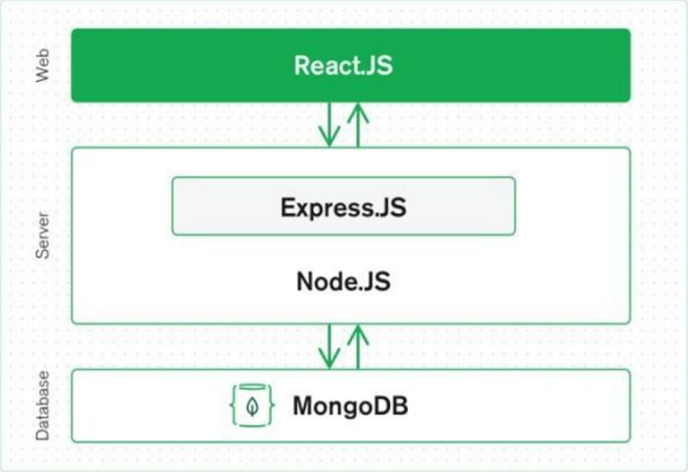


Fig 3.1: A 3-tier MERN architecture

B. *System Design Implementation*

The system is designed following the principle of separation of concerns by dividing the application into frontend and backend modules [2]. The client-side is responsible for user interaction, interface rendering, and user experience, while the server-side handles authentication, data processing, CRUD operations, and database management [11].

The major features of the system include:

- User registration and secure login
- Profile image upload
- Profile management
- Post creation, editing, and deletion
- Likes, comments, and social interaction
- Friend management system

- Responsive design
- Light and dark mode support

This modular implementation improves maintainability and allows efficient development across different system layers [13], [15].

1) Technical Requirements

The development environment requires:

- Visual Studio Code (IDE)
- Postman (API testing)
- MongoDB Database
- Node Package Manager (NPM)

Table 3.1: Hardware Configuration

Hardware Component	Specification
Processor	AMD Ryzen 7 3750H @ 2.4 GHz
RAM	16 GB
System Type	64-bit Windows 11

Table 3.2: Software Configuration

Software Component	Specification
Operating System	Windows
Programming Language	JavaScript / JSX
Package Manager	NPM
Runtime Environment	Node.js
Frontend Framework	React.js
Backend Framework	Express.js
Database	MongoDB

C. Front-End Development

The frontend of Nexora is exclusively developed using React.js, enabling reusable component-based architecture and dynamic rendering through the Virtual DOM [13]. React Router manages application navigation, while Formik and Yup provide robust form validation. Redux Toolkit with Redux Persist is used for centralized state management, ensuring efficient data flow throughout the application [1], [2]. Material UI improves visual design, and React Dropzone enables media uploads.

This frontend architecture enhances:

- Scalability
- Performance
- Reusability
- User experience
- Responsive design [14], [16]

D. Back-End Development

The backend infrastructure is implemented using Node.js and Express.js [9], [11]. Node.js provides an asynchronous runtime environment optimized for server-side execution, while Express.js simplifies API development and middleware integration [4].

Core backend functionalities include:

- User authentication using JWT
- Secure API communication
- CRUD operations
- Image uploads via Multer
- Route management
- Middleware security
- Database integration

MongoDB is used as the primary NoSQL database, while Mongoose manages schema design, validation, and database interactions [6], [7], [8]. This backend structure ensures secure communication, optimized performance, and scalable deployment [18].

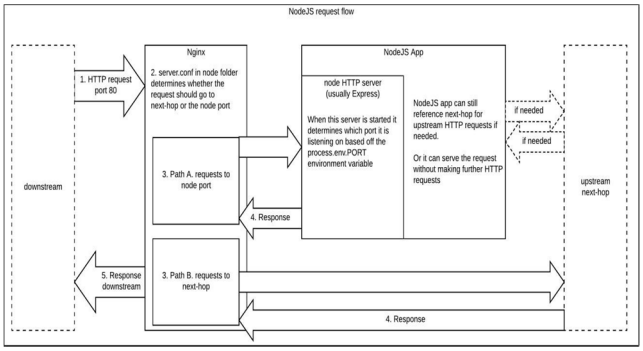


Fig 3.4: NodeJS Request Flow

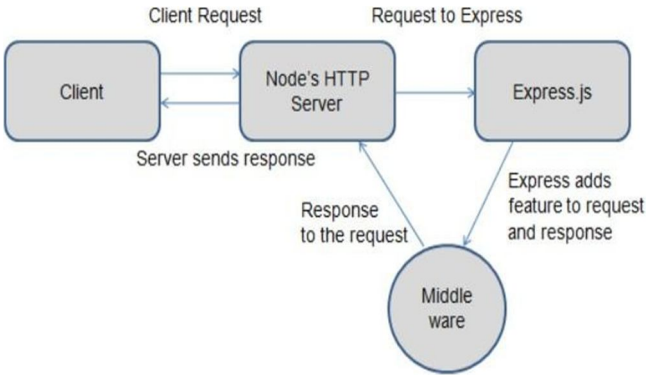


Fig 3.5: Express Js Request Flow

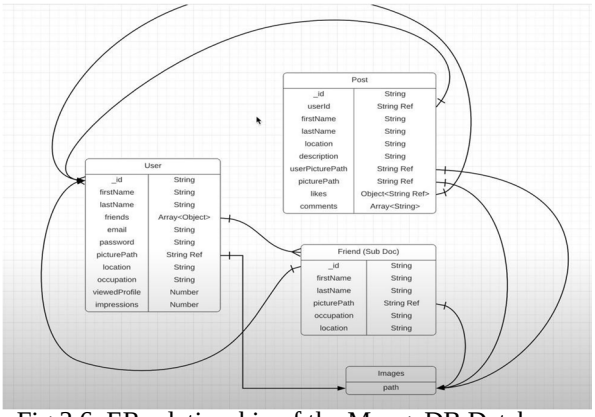


Fig 3.6: ER relationship of the MongoDB Database

IV. RESULT AND ANALYSIS

A. Login and Registration

The Nexora platform successfully implements secure user authentication through validated registration and login functionalities [18]. New users can create accounts by providing personal details, profile images, and authentication credentials through a fully validated registration form. User information is securely transmitted to the backend server in JSON format and stored in the MongoDB database using encrypted authentication mechanisms [6], [7]. Each registered user is assigned a unique system identifier, ensuring secure profile management and protected access.

The authentication system verifies registered credentials before granting platform access, ensuring security, data integrity, and seamless user experience [9], [18].

B. Homepage Analysis

The homepage provides a responsive and interactive social networking environment featuring social feeds, user profiles, friend lists, and post management functionalities. Users can create posts, interact with content through likes and comments, manage friends, and personalize their experience using light and dark mode features [15], [16].

The responsive UI design improves accessibility across multiple devices while maintaining scalability and performance [13].

C. Post Creation and Social Interaction

The system enables users to create, edit, and delete posts with support for media uploads, descriptions, and metadata storage. Each post is stored in MongoDB along with relevant fields such as timestamps, user credentials, and social interaction data [6], [7].

Core post functionalities include:

- Post creation
- Post editing
- Post deletion
- Likes and dislikes
- Comments
- Friend interactions

This modular implementation ensures efficient rendering, maintainability, and real-time social engagement [1], [2].

The platform supports comprehensive post management by allowing users to create, modify, and remove content with media uploads, descriptive details, and structured metadata storage. MongoDB efficiently maintains post records, including timestamps, user credentials, and social interaction information, ensuring secure and scalable data management [6], [7].

Table 4.1: Functional Testing Results

Feature	Status	Result
User Registration	Passed	Successful
User Login	Passed	Secure
Profile Management	Passed	Functional
Post Creation	Passed	Functional
Post Editing/Deletion	Passed	Functional
Likes & Comments	Passed	Functional
Friend Management	Passed	Functional
Responsive Design	Passed	Optimized

D. Performance Analysis

The system was evaluated through coding standard validation, linter testing, API response verification, and frontend responsiveness analysis [13], [14]. Linter checks confirmed adherence to industrial coding standards with no major formatting or structural issues.

Performance analysis indicates:

- Secure authentication
- Fast API communication
- Stable database operations
- Responsive UI rendering
- Efficient scalability

Table 4.2: Performance Metrics

Metric	Evaluation
Authentication Security	High
API Response Time	Fast
Database Performance	High
UI Responsiveness	Excellent
Mobile Compatibility	Fully Responsive
Scalability	Strong
Code Maintainability	High

E. Dependency Management

The Nexora platform utilizes structured dependency management through package configuration files, ensuring stable software version control and maintainable development environments [2]. Both frontend and backend dependencies are managed systematically to support deployment consistency and future scalability.

F. Mobile Responsiveness

The platform demonstrates complete responsiveness across desktop, tablet, and mobile devices. Adaptive layouts, scalable components, and responsive styling ensure seamless user experiences regardless of screen size [15], [16].

V. CONCLUSION

The primary objective of the Nexora project was to understand, implement, and integrate modern full-stack web development technologies, specifically React.js, MongoDB, Express.js, and Node.js, in order to develop a scalable social networking platform capable of performing secure CRUD operations through a structured three-tier architecture [1], [2]. The project successfully demonstrates the practical application of MERN stack technologies in designing and deploying a responsive, secure, and maintainable web application that fulfills modern social networking requirements.

Throughout the development process, the project explored the core principles, architecture, and practical implementation of each MERN stack component, including frontend development, backend services, API integration, authentication mechanisms, and NoSQL database management [3], [4]. The successful integration of React.js for frontend responsiveness, Node.js and Express.js for server-side execution, and MongoDB for dynamic data storage highlights the effectiveness of JavaScript-based unified full-stack development.

Nexora successfully implements essential social networking functionalities such as secure user authentication, profile management, post creation, social interaction, friend management, responsive design, and mobile compatibility. These features collectively demonstrate the practical viability of MERN stack architecture for solving real-world web application challenges [5].

The project also provided valuable insights into software engineering best practices, including clean code structure, modular development, reusable components, scalability, performance optimization, and user-centered interface design [13], [15]. It emphasizes that successful software development extends beyond technical implementation and must also prioritize usability, maintainability, and future extensibility.

Overall, the project can be considered successful as it achieved all major objectives within the defined scope and time constraints. Nexora serves as an effective demonstration of how MERN stack technologies can be utilized to create real-world, industrial-grade social networking systems that are scalable, secure, and adaptable for future enhancements [4], [19].

Future improvements may include:

- Real-time chat integration
- Advanced AI-based recommendations
- Enhanced analytics dashboards
- Improved deployment optimization
- Expanded security protocols

The Nexora project not only strengthened practical full-stack development skills but also reinforced the importance of structured software engineering, responsive design, and continuous innovation in modern web application development.

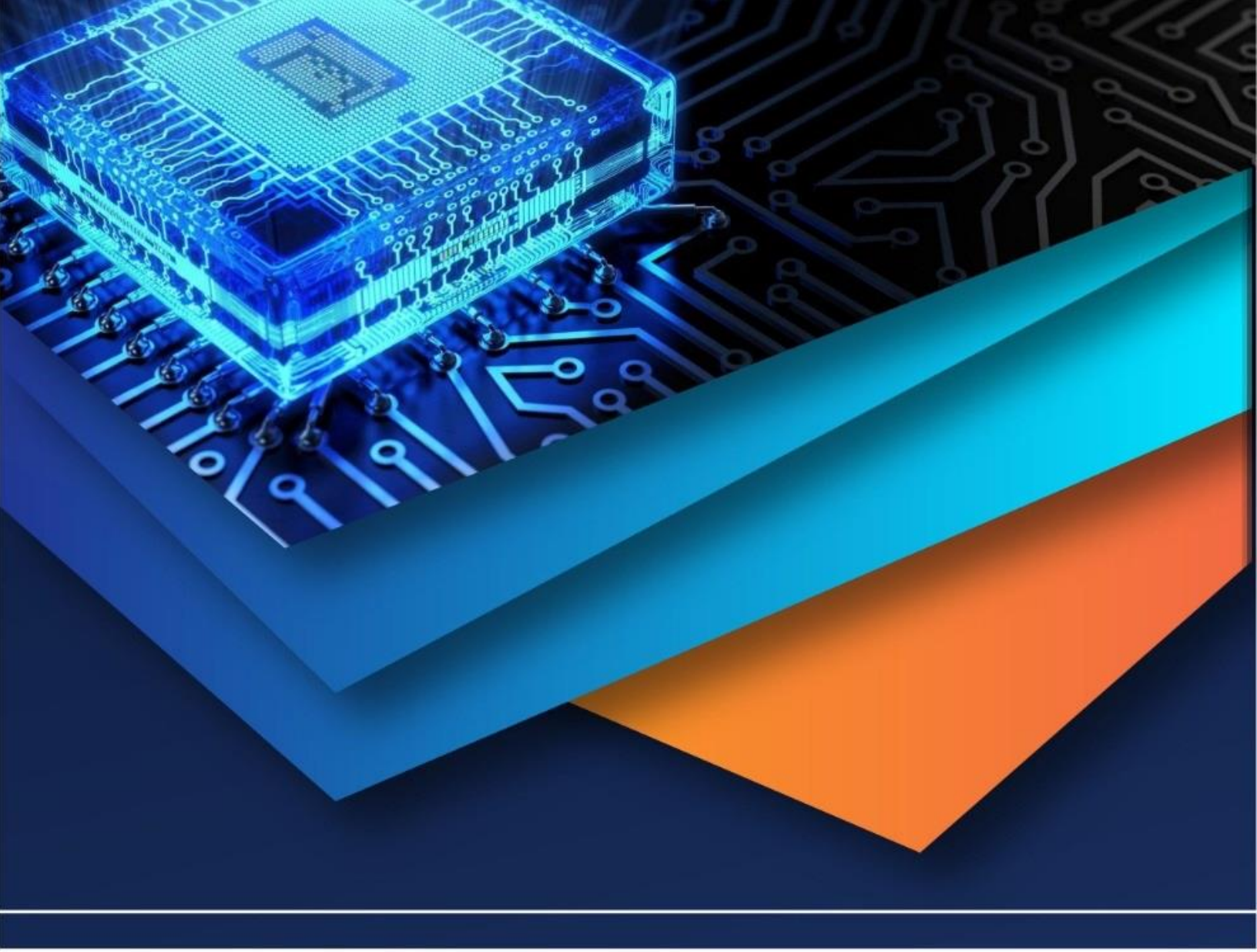
VI. ACKNOWLEDGEMENT

We sincerely express our gratitude to our project guide, Prof. Vishal R. Deshmukh, for his valuable guidance, continuous support, and encouragement throughout the development of this research work. We also thank the Department of Computer Engineering and Shri Shivaji Vidya Prasarak Sanstha's Bapusaheb Shivajirao Deore College of Engineering, Dhule, for providing the necessary resources and support.

REFERENCES

[1] S. M. Malewade and A. Ekbote, “Performance Optimization using MERN Stack on Web Applications,” IJRASET, vol. 10, 2021.
[2] Y. Baiskar, P. Paulzagade, K. Koradia, P. Ingole, and D. Shirbhate, “MERN: A Full Stack Development,” IJRASET, 2022.
[3] A. Singh and A. Anikesh, “Web Development and Computer Science and Engineering,” IJRASET, 2022.

[4] P. Kaushik, S. Suman, B. D. Shivahare, and V. Bibhu, "Web Development and Performance Comparison of Web Development Technologies in NodeJS and Python," IEEE ICTAI, 2021.
[5] P. D. Dutonde, "Website Development Technologies: A Review," IJRASET, 2022.
[6] M. Stonebraker, "SQL Databases v. NoSQL Databases," Communications of the ACM, 2010.
[7] S. H. Aboutorabi et al., "Performance Evaluation of SQL and MongoDB Databases," IEEE CSSE, 2015.
[8] C. Chodorow, "Introduction to MongoDB," FOSDEM, 2010.
[9] S. Tilkov and S. Vinoski, "Node.js: Using JavaScript to Build High-Performance Network Programs," IEEE Internet Computing, 2010.
[10] A. Boicea et al., "MongoDB vs Oracle—Database Comparison," IEEE, 2012.
[11] L. Liang et al., "Express Supervision System Based on NodeJS and MongoDB," 2017.
[12] M. R. Solanki and A. Dongaonkar, "A Journey of Human Comfort: Web1.0 to Web4.0," IJRSI, 2016.
[13] A. Javeed, "Performance Optimization Techniques for ReactJS," 2019.
[14] J. M. Spool, "Content and Design are Inseparable Work Partners," 2014.
[15] H. Bozikovic and M. Stula, "Web Design: Past, Present and Future," MIPRO, 2018.
[16] B. Carter, "HTML Architecture, a Novel Development System (HANDS)," 2014.
[17] A. Sterling, "NodeJS and Angular Tools for JSON-LD," IEEE, 2019.
[18] D. Laksono, "Testing Spatial Data Deliverance in SQL and NoSQL Database Using NodeJS Full Stack Web App," 2018.
[19] M. M. Patil et al., "Performance of MongoDB vs MySQL Database," 2017.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY